Performance and Use Evaluation of an Electronic Book for Introductory Python Programming

Christine Alvarado
Harvey Mudd College
Claremont, CA
christine.alvarado@gmail.com

Briana Morrison, Barbara Ericson, Mark Guzdial
College of Computing
Georgia Institute of Technology
801 Atlantic Drive
Atlanta, GA 30332-0280
[bmorrison, ericson, guzdial]@cc.gatech.edu

Brad Miller, David L. Ranum
Luther College
700 College Dr.
Decorah, IA 52101
[bmiller,ranum]@luther.edu

ABSTRACT

Electronic books (ebooks) provide the opportunity to go beyond the limitations of a physical page. These opportunities are particularly important for computing education, where dynamic information is a key characteristic of our domain. An electronic book can provide opportunities to program or conduct analyses that are impossible on the physical page, integrating instructional information with creative exploration. However, just because ebooks provide these opportunities does not mean that we know how students will actually use ebooks in the context of a class. Miller and Ranum have produced an electronic book for teaching introductory computing in Python. We explored how students used the dynamic and novel features of the book, and correlated that use with performance on learning measures. We found that students made extensive use of the traditional programming environment in the book, but that the lesser-used visualization tool was better correlated with student performance. In addition, we found that although students reported high levels of satisfaction with the book, they appeared to use it much like a traditional textbook, making less use of many of the interactive features of the book than we expected.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education — computer science education, information systems education

General Terms

Measurement, Documentation, Design, Experimentation, Human Factors

Keywords

electronic books, ebooks, distance learning, performance, assessment, usability

1. THE PROMISE OF ELECTRONIC BOOKS

Learning computer science has always been hard. We know that one of the sources of this difficulty is cognitive overhead [4]: students must learn unique problem-solving skills while at the same time learning a new notation for implementing solutions, i.e., programming languages [1]. Reducing the cognitive overhead of learning to program might make it easier and allow more students to be successful.

There are instructional methods for reducing cognitive overhead [7]. For example, we know that helping students to attend to the most important issues in learning by focusing their attention is critical to improving learning [2]. Traditional instructional materials (e.g., textbooks, lecture notes, instructional videos, visualization tools) require students to attend to many different environments and might lead to the kind of split attention problem that can hurt students' learning [1].

We also know that a cause for failure in introductory computing is a sense of low self-efficacy, e.g, students believe that they just are not "computer people," and that struggles with setting up tools and IDE's contribute to that sense of low self-efficacy [3]. Reducing the number of tools and the number of installations might not just reduce cognitive overhead. Fewer installs means fewer opportunities for failure and fewer examples to reinforce negative self-efficacy.

New kinds of electronic books (ebooks) for computer science that integrate tools and other learning resources provide an opportunity to reduce the cognitive overhead and challenges to self-efficacy. For example, Guido Rössling has created a computer science textbook that integrates instructional materials with algorithm visualization, using Moodle which includes the possibility of integrating social media (e.g., chats) and problem-solving exercises [6]. Integrated electronic textbooks can provide a single interface and structure for a wide variety of instructional materials. However, these tools do not reduce the single greatest source of cognitive overhead and challenge to self-efficacy perceptions – the programming language and IDE.

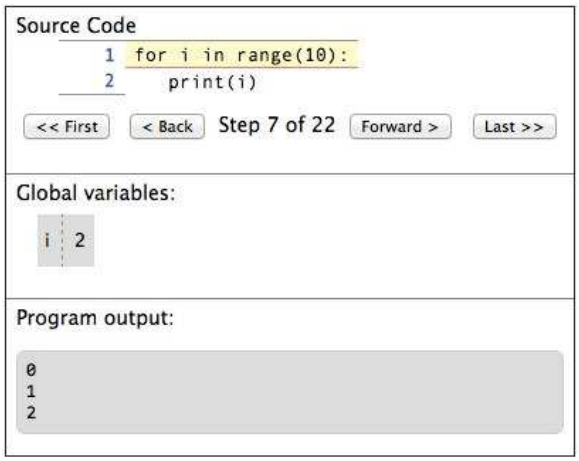

Figure 1: An example codelens visualization

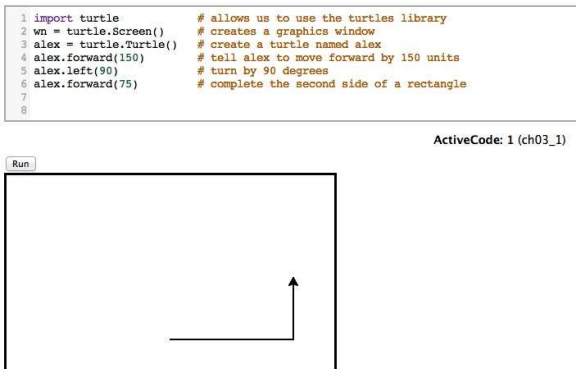

Figure 2: An example activecode execution area

Brad Miller and David Ranum have produced a new electronic book for computer science that integrates a textbook with a variety of other instructional materials, including a programming environment [5]. The book offers three new features:

- Embedded instructional videos.
- Codelens, a software visualization tool (Figure 1). Using codelens students can step through code line-by-line, forward or backward, or all the way to the start or end of the program execution.
- Activecode, a program editing and execution area (Figure 2). Using activecode students can execute examples, change them, and execute the changed code. They can also save and load their modified code.

Miller and Ranum's book gives us a chance to see how student behavior might change (or not) with these books compared to traditional tools, and what might be the impact of this integrated resource on learning.

We studied the first semester of use of the Miller and Ranum textbook in the introductory computer science course at Luther College during Fall 2011. We created measures of student performance, completely separately from the authors/teachers. We analyzed student behavior from studying log files, then correlated feature use with student performance. Finally, we surveyed students to understand their thoughts about the ebook.

Based on this research and analysis, we present three central contributions to the understanding of ebooks for learning computer science. First, we establish a baseline for student use of this type of electronic learning resource against which to measure future use of this and other electronic learning resources and opportunities. Although it would be interesting to examine how students' use of an integrated electronic textbook differs from their use of a traditional textbook with supplementary tools, there exists little data on how students actually use traditional textbooks. For example, we expected more use of the books' novel features than we saw, yet we acknowledge that these expectations were based on our own intuitions rather than on theory or other studies. Second, we find that although there was low average use of the code visualization tool, the variance in amount of use was large between students, and this tool was most correlated with performance outcome. Third, student responses to our survey indicate that although they enjoyed the ebook, their approach to studying with it was not much different from their approach to studying with a traditional textbook.

2. METHOD

Our subjects were the students in Miller and Ranum's two sections of introductory computing at Luther College. Our experimental protocol was reviewed by both Georgia Tech's and Luther College's human subjects review boards. There were a total of $n = 61$ students across the two sections.

There were four sources of data in this study:

- The team from Georgia Tech[1] created quizzes on chapters 2, 3, and 5. Chapter 2 covers variables, expressions and statements; chapter 3 covers basic programming using for-loops via the `turtle` package; chapter 5 covers writing and calling functions. The quizzes were developed without any oversight from the author/teachers, based solely on the material in the book. Our aim was to measure performance based on content in the chapter, without regard for what was emphasized in lecture. Each quiz was administered in lecture, shortly after the students had completed each chapter. All three quizzes were unannounced, did not count toward students' grades, and were completed before the first midterm examination.
- Miller and Ranum provided the Georgia Tech team with the results of the first midterm examination.
- We had access to log files describing student behavior in using the electronic book (Figure 3).
- At the end the course, we asked the students to complete a survey.

[1]This team includes Christine Alvarado, who is visiting Georgia Tech in 2011-12.

```
save|ex_2_12|activecode|m@l.edu|2011-08-31 03:01:32.490527
play|inputvid|video|r@l.edu|2011-08-31 03:23:10.533168
play|reassignmentvid|video|r@l.edu|2011-08-31 04:07:51.518135
save|ex_2_4|activecode|m@l.edu|2011-08-31 13:16:55.379646
run|ch02_1|activecode|m@l.edu|2011-08-31 13:17:06.192521
```

Figure 3: A fragment of a log file

The format of each statement in the log file is:

```
action | identifier | feature | user | timestamp
```

For example, the first line in Figure 3 means that "on 2011-08-31 03:01:32.490527 the user m@l.edu pressed the save button on an activecode block with the id of ex_2_12."

There were three features identified in the log file:

- *codelens*: Valid actions for codelens visualizations were: first, last, fwd, and back.
- *activecode*: Valid actions for activecode events were: save, run, and load.
- *video*: The only valid activity for video was to play.

We analyzed the log files by counting features and activities, by student, over time. We chose activity counts instead of use time because we felt these counts would be the most accurate measure of student engagement with the tools. In particular, trying to estimate time on task by measuring time between logged activities would fail to take into account time students spend multitasking and might vastly over-estimate tool engagement for some students. In addition, we characterized the log file events as during class (when Brad and David were directing effort) and outside of class. We correlated those counts with performance on the quizzes and midterm exam to determine what features most highly correlated with high performance.

On the final survey, we asked students to tell us what they felt was useful in their learning, what their strategies were for studying in this course, and about their experiences with e-readers. We asked students to rank the learning resources in the class in terms of how much each helped in learning. We asked students about their experiences reading physical or electronic books, and how they much read for different kinds of classes.

We used the survey results to provide insight into the behavior we saw in the log files. The survey included a few questions with text answers, such as what additional resources students used for learning, what students did to study, and what they thought should be changed about the course. We did not analyze the text written by the students, and we did not correlate final survey results with log file or student performance data.

3. RESULTS

Key to understanding the results is that students had lecture with computers in front of them. The focus in lectures was not on repeating the exercises and activities book, but to do things *not* in the book. Lecture engaged students in exercises brought from outside. The students often used one of the scratch editors in the book, or teachers chose one of the existing book exercises to have students work as part of a classroom activity. Outside of class, students had regular reading assignments, and while the teachers encouraged the students to work with activecode and codelens, students were not required to use those features. After the midterm, students shifted to use PyCharm as their IDE.

Here are the first three out-of-class programming assignments from the class:

- Do Problem 12 in Chapter 2 and make sure you save it! Experiment with using the input statement.
- Write a program that will convert degrees fahrenheit to degrees celsius.
- Use the turtle to draw a picture of anything you like. Be creative. Experiment with different turtle methods.

Throughout this section we present results using log data for the first half of the semester. We do not include the second half for two reasons. First, all of our learning measures took place in the first half of the course. Second, midway through the course students began using PyCharm as their IDE, and we wanted to focus on the portion of the course where students were working within a single unified tool. Indeed, use of the active portions of the book declined when students began using PyCharm.

3.1 Performance on learning measures

Performance on the quizzes created by the Georgia Tech team and on the midterm examination created by the teachers at Luther College is summarized in Table 1. Performance was good, and the scores had enough variance among the students that we felt that log file analysis might provide us some insight into differences. We noted that the midterm examination grade was higher than the average of the quiz grades with low standard deviation. The midterm examination might have been a better fit for the focus of the course, and students might have taken the examination more seriously than the quizzes. For one, it is likely that students explicitly studied for the midterm, but did not study for the quizzes because the quizzes were unannounced and did not count toward their grades.

Table 1: Student performance on quizzes and midterm examination. Scores normalized so that 1.0 would be 100%.

	Average (Standard deviation)
Quiz Chapter 2 (6 problems)	0.74 (0.20)
Quiz Chapter 3 (9 problems)	0.83 (0.14)
Quiz Chapter 5 (9 problems)	0.47 (0.24)
Midterm Examination	0.83 (0.15)

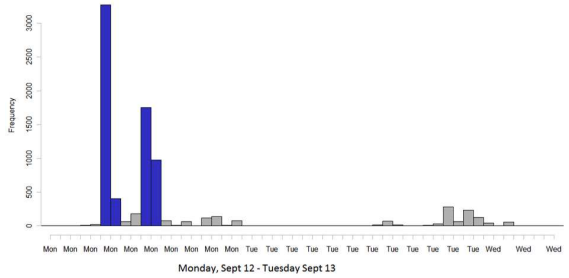

Figure 5: Histogram of all student use (by event) for just two days in second full week of class. Blue bars show class times.

3.2 How students used the ebook

More interesting for us was understanding how students used the electronic book. Figure 4 shows the cumulative number of events (of all types) per day. Use on days with class are blue, use on other days are gray, and the midterm date (30 September) is in red. Class started on 31 August, so Monday 5 September is the start of the first full week of school, which is marked with an arrow.

Overall, we see a lot of use whose amount varies substantially between class days and non-class days. Cumulatively, out-of-class use was slightly higher overall than in-class use: Over the course of the whole semester, there were 121, 318 in-class logged events, and 114, 046 out-of-class logged events (and over the period of time covered by Figure 4 there were 68, 571 in-class logged events and 78, 477 out-of-class logged events). However, the relatively short time students were in class leads to high peaks in activity during class time. Figure 5 zooms in for an hour-by-hour analysis of use during two days in the second full week of class including one class day and one non-class day. Here we can see the high use of the book's tools specifically during class time.

When do students use the ebook? As we have already noted, there is significant use in class. Students also use the book immediately after class and in the evening hours on the days before class, as well as on the weekend. These usage patterns match the typical studying we expect students to engage in, though at a lower rate than we had predicted. We had expected that students would spend several hours studying for every hour they spend in class resulting in much greater out-of-class use. (Though we note that we do not have reliable comparison data for traditional study habits.) In addition, we expected to see a significant number of events just before the midterm exam, as students studied for the midterm. Figure 6 describes those events. While we do see an increase in activity before the midterm, this activity is swamped by the class-day use just a few days before.

Students did not use the features of the book equally. Figure 7 depicts activecode area events. Figure 8 describes all codelens events. Figure 9 describes all video events. Activecode events (i.e., traditional coding activities) account for the vast majority of logged events. With one or two exceptions, codelens events decline more sharply than activecode events over the course of the semester, but continue to be

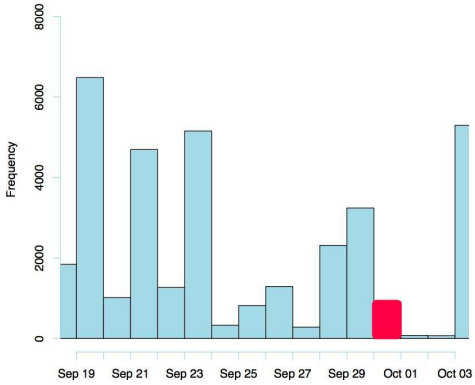

Figure 6: Histogram of all student use for the days preceding the midterm exam. Red indicates the day of the exam.

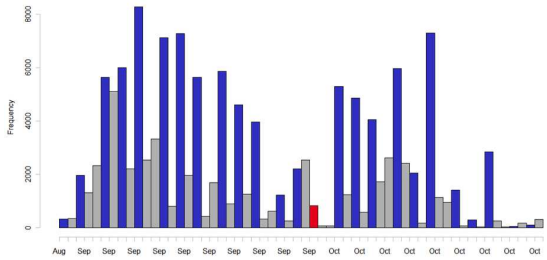

Figure 7: Histogram of all student use of Activecode.

present throughout the time period we analyze. Video use declines sharply, and all but disappears after the midterm exam.

Finally, *who* is using the ebook features? Use varies dramatically among the students. Consider Figure 10 which is a histogram describing the number of students (height) sorted into bins of how much use they make of the special ebook features (logged events) outside of the class time. We see that no one is in the zero bin – all students make *some* use of the tools. Most students log between 100–500 activities in the course of the half semester. However, the distribution has a heavy tail: There are a few students who use the tools a lot more than the majority of the students.

Another way to get a sense of the broad distribution of use is to look at what fraction of the students use the tools *at all* outside of class over the course of the semester. As mentioned earlier, video use was fairly insignificant after the first few weeks of the semester. We saw that students do use the activecode and codelens facilities, but not all students to the same extent. Table 2 presents the percentage of students who used activecode or codelens *even once* outside of class in each of the weeks preceding the midterm exam[2]. For

[2]Class started on 31 August, but all weeks start on Monday here. Weeks start at 5am Monday and end at 4:59am on the following Monday. The last week is an exception where activities cut-off at 1:30pm for the midterm exam.

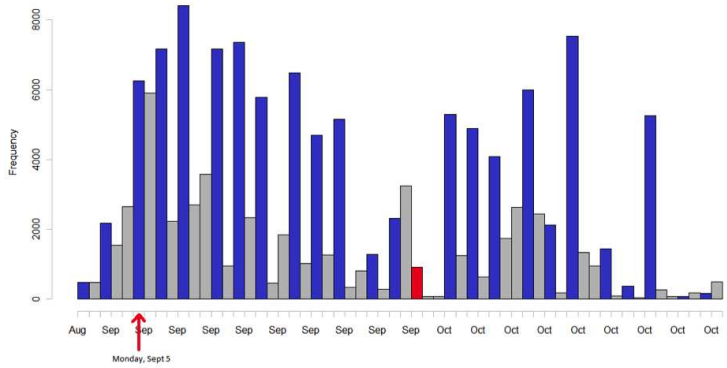

Figure 4: Histogram of all student use (by event) of the interactive pieces of the book, by day. Blue bars show class days. The red bar is the day of the midterm.

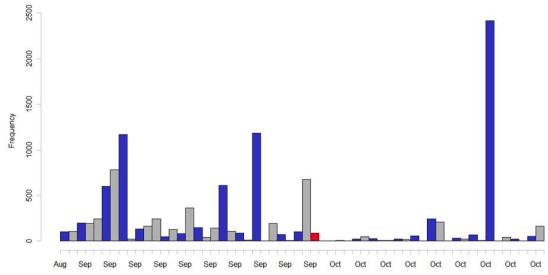

Figure 8: Histogram of all student use of Codelens.

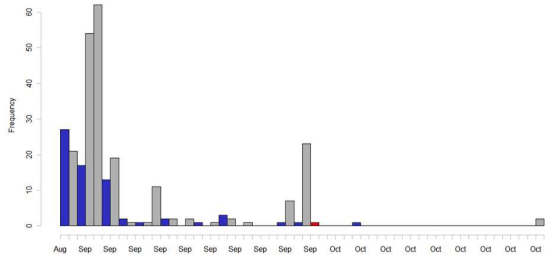

Figure 9: Histogram of all student use of Videos.

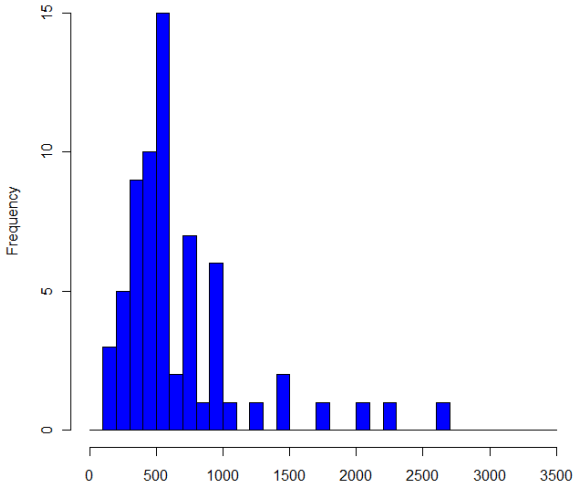

Figure 10: Bins on x-axis are number of logged events outside of class, and y-axis represents the number of students in each bin.

example, only 18% of students used codelens outside of class in the week leading up to the midterm (i.e. to study for the exam). Codelens use falls off dramatically, while activecode use stays relatively high (though 17% of the students did no programming in activecode outside of class to study for the midterm exam).

We also looked at what fractions of students accounted for what percentage of tool use outside of class. We defined three groups: high-users (top 1/3 of codelens and activecode users, with only moderate overlap), medium users (middle 1/3 of each), and low users (bottom 1/3 of each). The groups overlap between codelens and activecode only about 50%. That is, only about half the users in the low codelens group are also in the low activecode group. For activecode, the top 1/3 of users accounted for about 57% of activecode use, the middle 1/3 accounted for about 27%, and the bottom 1/3 accounted for about 16%. For codelens, the results were similar, though the low bin accounted for even less use: The top 1/3 of users accounted for about 58% of codelens use,

Table 2: What percent of students use Activecode and Codelens in each week?

	Activecode	Codelens
Week 1 (Aug 29–Sept 4)	97%	70%
Week 2 (Sept 5–Sept 11)	100%	55%
Week 3 (Sept 12–Sept 18)	100%	36%
Week 4 (Sept19–Sept 25)	92%	14%
Week 5 (Sept 26–Sept 30)	83%	18%

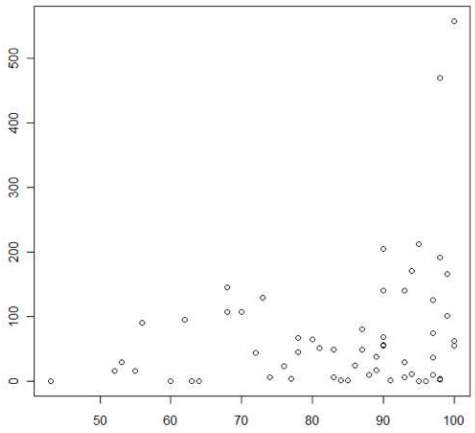

Figure 11: Scatterplot of midterm score along x-axis, and number of codelens events along y-axis.

the middle 1/3 accounted for about 33%, and the bottom 1/3 accounted for about 9%.

3.3 Correlating use with performance

The only event correlated with midterm grade performance was the students' use of codelens outside of class. We found that the correlation between codelens use *outside of class* and midterm grade was $r = 0.27$ ($p < 0.05$). A scatterplot of midterm grade and codelens use shows the relationship (Figure 11).

Including codelens use *during class* weakens the correlation and makes it non-significant. The use of the video or the activecode tool, whether inside or outside of class, was not significantly correlated with students' performance on the midterm or on the quizzes.

For all three quizzes, we see a positive correlation between students' total quiz score and their codelens use outside of class, but only the correlation for Quiz 2 is statistically significant with $r = 0.31$, $p < 0.05$. (Chapter 3 Quiz: $r = 0.21$, $p = 0.12$ and Chapter 5 Quiz: $r = 0.25$, $p = 0.64$.)

Looking at the graph for Chapter 2 (the other two look similar), we can see the same positive trends we saw for the midterm exam (Figure 12). Because these quizzes were unannounced, we believe that students' regular use of the codelens tool (rather than their use in specifically studying

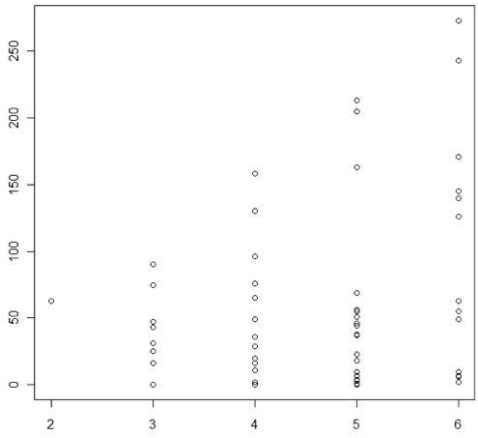

Figure 12: Scatterplot of quiz 2 along x-axis, and number of codelens events along y-axis.

Table 3: Most useful learning resources by student self-report. Sorted by rating average

Resource	How often picked #1	Rating average
Lectures	22	2.0
Running programs from the textbook	11	3.37
Reading the textbook	4	3.75
Changing programs and running the new programs in the textbook	9	3.81
Tracing programs using CodeLens	1	5.4
Watching videos in the textbook	4	5.91

for an exam) is what provides the most learning gains.

We hypothesized that the use of the 'back' option in the codelens, in particular, would more strongly correlate with students' performance on the midterm exam (and hence their understanding). Going 'forward' is stepping through the code, but going 'back' might suggest investigating *how* a certain program state was reached. While we did find that students used the back option much less than they used forward, we found that the use of both actions matched the correlation of use of the codelens tool overall. In other words, we did not see any stronger correlation between score and use of the back button than we did between score and any use of the codelens tool outside of class.

3.4 How students thought about the ebook

Students liked the books and its unique features. Table 3 describes students' ($n = 61$) ratings of the learning resources, in terms of how often students picked the resource the most useful ("#1" in rating), and the average score (where lower is most useful). Lectures were by far the most valued learning resource. Tracing programs with codelens was not highly ranked, but it is ranked higher than videos.

Table 4 describes student responses to Likert-scale questions. Students were positive about the class and about their ability to program after the course. They were quite

positive about the book, giving it the highest average score. Students were experienced reading books on the screen, but generally preferred physical, paper books. Students reported that they were likely to do the reading in humanities courses, but less likely to do the reading in STEM classes.

One of the final survey questions was "What did you do to study for your midterm exam in this Computer Science class?" ($n = 60$). There was an emphasis in the responses on "reading" and "looking." Only three of the students mentioned "code" (including references to activecode or codelens). Six students mentioned looking at "programs."

In response to the question, "What was your favorite thing about this class?," 22 of the 60 students mentioned the textbook. In response to the question, "What was your least favorite thing about this class?" only 4 of the 55 responses mentioned the textbook, but not always in an unfavorable way, as in this example: "I got bored in class. The profs tried to find ways to add extra things to do for the quick students, but this could be more developed. Maybe add more exercises to the textbook." These answers suggest that the ebook may have been reducing the kind of discouragement reported in dealing with installation of IDEs that can impact student perception of self-efficacy [3].

4. DISCUSSION

Overall, it is clear that the ebook was a success with the Luther College students. Students liked the book very much. Student performance on quizzes and tests were good. Performance on the midterm and one of the quizzes correlated with use of a unique feature of the book.

The correlation between codelens use and some performance measures is not causal evidence, of course. There are several possible explanations for the correlation. Did students perform better in the course than they might have because they had access to codelens? Or did the better students tend to use codelens? We can't know from these results, but instead, they raise new questions.

Why was there such an enormous variance in student use of the features? The variance poses a challenge in designing future versions of electronic books for computer science. How would we characterize these users? Are there features that we should be providing for the heavy users, or are there missing features that would make the tool more valuable for the light users?

Is it surprising that the Luther College students did not use the textbook and its facilities more out-of-class? The Georgia Tech team was expecting to see two or three hours of programming for each hour of lecture (or two or three times more events out-of-class than in-class), but we didn't see that. Perhaps the close match of in-class and out-of-class use is exactly what we could see if we could actually measure use of the textbook and IDE in a traditional course. Perhaps only a few students *ever* read the book at all outside the class. In fact, the Luther College students might actually be working outside of class *more* than in a traditional class. We can only see a difference here because use of the book is logged.

In general, there was less use of the unique features of the ebook than we had originally anticipated. Since the IDE was in the book, we expected to see hours of additional programming for every hour of in-class lecturing. Similarly, the lack of use of the videos was quite surprising. Videos are core to many distance education efforts (e.g., Kahn Academy, Coursera, Udacity, and MITx), and students seemed to like the videos. Why didn't they use them more?

One possible explanation for the less-than-anticipated use of the unique features may be student study skills. The survey results tend to suggest that students "study" by "reading." Few students mention coding or tracing programs as a way of "studying" computer science. Might students have learned more if we encouraged them to use codelens more? We may need to teach students new study skills to take advantage of new learning resources and opportunities.

5. CONCLUSIONS AND NEXT STEPS

Our evaluation of the Luther College ebook suggests that it was a significant success. We saw no indication of students suffering distress over battling with installation of languages or IDEs in the course. We saw many indications of successful use, of student enjoyment of the book, and of learning opportunities.

A clear next step is to conduct a more comparative study. How does learning with the ebook differ from a course with more traditional learning opportunities? A challenge to conducting a comparative study is getting similar data from a more traditional class. The ebook gives us the opportunity to see into student behavior in ways that are challenging to do otherwise. Was there less use of the ebook than a physical book? How could we instrument a physical book to know?

Another interesting step would be to add some meta-instruction. Can we teach students new study skills, to take advantage of the unique resources of the book? New media may demand a change in how students use the media.

Both the Georgia Tech and Luther College teams are continuing their exploration of electronic books. We are collaborating on embedding more kinds of interactive problems for students to use in the book, such as multiple choice questions. Our goal is to drive use of the book and its unique features, by giving students more opportunities to check their understanding. If we can give students feedback, to make clear what they do and don't understand, they might realize the value of the interactive features in improving their understanding and increase their use of the features.

At Luther College, Brad and David are exploring how to add some of the additional features that students have requested for the book. For example, students would like to be able to highlight passages and make notes in the book. These features give students access to some of the personalization and learning opportunities that are available in physical books.

At Georgia Tech, we are interested in using these books in a new context, with high school teachers who are learning computer science at a distance. Our teacher-students will not have the advantage of a lecture. They would be using

Table 4: Student responses where 1 is "That describes me perfectly" and 5 is "I can't even imagine being like that." Ordered as presented to the students.

I prefer to read textbooks in physical paper than on a screen.	2.85
I read books or similar documents (e.g., PDFs) on the screen regularly.	2.37
I like to read books on electronic readers like Kindle, Nook, or iPad.	3.31
I always do the required reading in classes like English or History.	2.45
I rarely do the reading in classes like Science or Mathematics.	3.44
I typically read the related readings before each lecture.	2.66
I now feel like I could write a Python program for something I would want to do.	2.06
I could write a Python program for a class (like in Mathematics) now.	1.95
Now that I have taken this class, I am planning to take another CS class.	2.66
I do not feel confident about programming in Python.	3.77
This class has made me less interested in Computer Science than I was before I started.	4.08
If I took another CS class, I would want to use a book like this one (e.g., with code examples in it).	1.24
It was easy to move to PyCharm after using the textbook for this class.	1.52
I could easily install and use PyCharm on a new computer.	1.89
I feel like I could program anything in Python.	3.31
I studied for this class pretty much like how I studied for any other class.	3.02

the ebook as their sole learning resource. Will the teacher-students use the book similarly, or would their use differ without the additional support of lecture?

Ebooks hold great promise for computer science education, and may help us to deal with some of our pervasive problems. This paper has provided a detailed view of one book's use and student performance. Our study can serve as a baseline, offering measures for others to use and as a comparison point for new ebooks.

6. ACKNOWLEDGMENTS

This paper is based upon work supported by the National Science Foundation under Grant CNS-1138378. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

7. REFERENCES

[1] J. Anderson, R. Farrell, and R. Sauers. Learning to program in lisp. *Cognitive Science*, 8(2):87–129, 1984.

[2] P. Chandler and J. Sweller. The split-attention effect as a factor in the design of instruction. *British Journal of Educational Psychology*, 62(2):233–246, 1992.

[3] P. Kinnunen and B. Simon. Cs majors' self-efficacy perceptions in cs1: results in light of social cognitive theory. In *Proceedings of the seventh international workshop on Computing education research*, ICER '11, pages 19–26, New York, NY, USA, 2011. ACM.

[4] P. A. Kirschner, J. Sweller, and R. Clark. Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, 41(2):75–86, 2006.

[5] B. Miller and D. Ranum. Beyond pdf and epub: Toward an interactive textbook. In M. E. Caspersen and O. Hazzan, editors, *17th Annual Conference on Innovation and Technology in Computer Science Education*, Haifa, Israel, 2012. ACM.

[6] G. Rössling and T. Vellaramkalayil. A visualization-based computer science hypertextbook prototype. *Trans. Comput. Educ.*, 9(2):11:1–11:13, June 2009.

[7] J. Sweller. Element interactivity and intrinsic, extraneous, and germane cognitive load. *Educational Psychology Review*, 22(2):123–138, 2010.